ICES REPORT 11-40

November 2011

Solid T-spline Construction from Boundary Representations for Genus-Zero Geometry

by

Y. Zhang, W. Wang, T. J. R. Hughes

The Institute for Computational Engineering and Sciences
The University of Texas at Austin
Austin, Texas 78712

Reference: Y. Zhang, W. Wang, T. J. R. Hughes, "Solid T-spline Construction from Boundary Representations for Genus-Zero Geometry", ICES REPORT 11-40, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, November 2011.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
NOV 2011		**00-00-2011 to 00-00-2011**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Solid T-spline Construction from Boundary Representations for Genus-Zero Geometry	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Texas at Austin,Institute for Computational Engineering and Sciences,Austin,TX,78712**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This paper describes a novel method to construct solid rational T-splines for complex genus-zero geometry from boundary surface triangulations. We first build a parametric mapping between the triangulation and the boundary of the parametric domain, a unit cube. After that we adaptively subdivide the cube using an octree subdivision, project the boundary nodes onto the input triangle mesh, and at the same time relocate the interior nodes via mesh smoothing. This process continues until the surface approximation error is less than a pre-defined threshold. T-mesh is then obtained by pillowing the subdivision result one layer on the boundary and its quality is improved. Templates are implemented to handle extraordinary nodes and partial extraordinary nodes in order to get a gap-free T-mesh. The obtained solid T-spline is C2-continuous except for the local region around each extraordinary node and partial extraordinary node. The boundary surface of the solid T-spline is C2-continuous everywhere except for the local region around the eight nodes corresponding to the eight corners of the parametric cube. Finally, a B?ezier extraction technique is used to facilitate T-spline based isogeometric analysis. The obtained B?ezier mesh is analysis-suitable with no negative Jacobians. Several examples are presented in this paper to show the robustness of the algorithm.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**24**	

Solid T-spline Construction from Boundary Representations for Genus-Zero Geometry

Yongjie Zhang [a,*], Wenyan Wang [a], Thomas J.R. Hughes [b]

[a] *Department of Mechanical Engineering, Carnegie Mellon University*
Pittsburgh, PA 15213, USA

[b] *Institute for Computational Engineering and Sciences, The University of Texas at Austin*
Austin, TX 78712, USA

Abstract

This paper describes a novel method to construct solid rational T-splines for complex genus-zero geometry from boundary surface triangulations. We first build a parametric mapping between the triangulation and the boundary of the parametric domain, a unit cube. After that we adaptively subdivide the cube using an octree subdivision, project the boundary nodes onto the input triangle mesh, and at the same time relocate the interior nodes via mesh smoothing. This process continues until the surface approximation error is less than a pre-defined threshold. T-mesh is then obtained by pillowing the subdivision result one layer on the boundary and its quality is improved. Templates are implemented to handle extraordinary nodes and partial extraordinary nodes in order to get a gap-free T-mesh. The obtained solid T-spline is C^2-continuous except for the local region around each extraordinary node and partial extraordinary node. The boundary surface of the solid T-spline is C^2-continuous everywhere except for the local region around the eight nodes corresponding to the eight corners of the parametric cube. Finally, a Bézier extraction technique is used to facilitate T-spline based isogeometric analysis. The obtained Bézier mesh is analysis-suitable with no negative Jacobians. Several examples are presented in this paper to show the robustness of the algorithm.

Key words: Solid T-spline Construction, Genus-zero Geometry, Rational T-spline, Isogeometric Analysis

1 Introduction

In solid modeling and Computer Aided Design, boundary representation is widely used in which a 3D "solid" geometry is represented using the boundary surface. Despite the necessity of boundary representation, interior volume data carries abundant information such as material properties and density. The whole solid model should be taken into account in many cases of solid modeling and physically-based analysis. For example, isogeometric analysis [10, 2], which utilizes NURBS

* Corresponding author: Y. Zhang. E-mail address: jessicaz@andrew.cmu.edu. Tel: (412) 268-5332; Fax: (412) 268-3348.

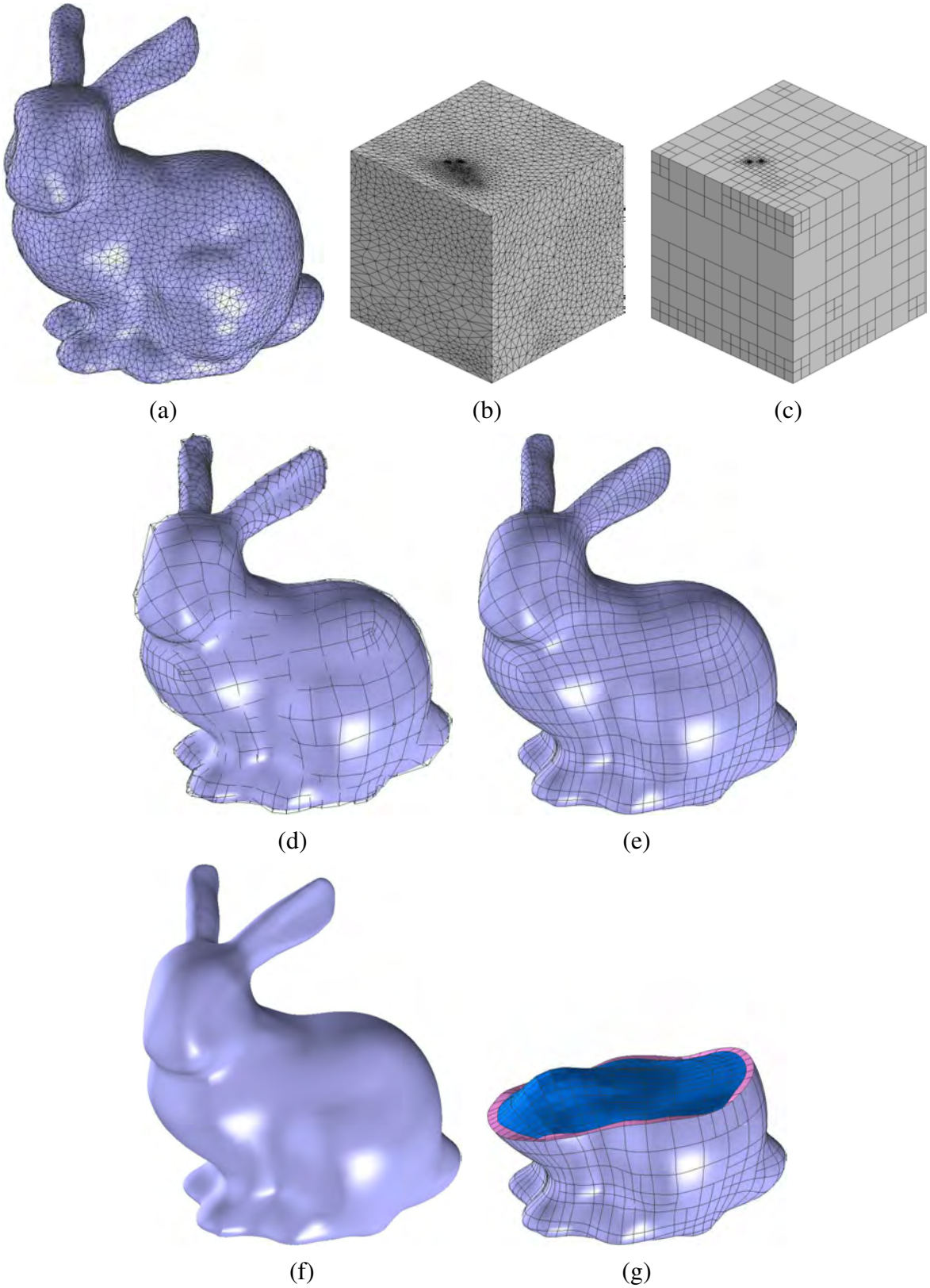


Fig. 1. Stanford bunny model. (a) The input boundary triangle mesh; (b) the mapping result; (c) the subdivision result for the parametric domain; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; (f) the solid T-spline; and (g) the extracted solid Bézier elements with some elements removed to show the interior mesh (blue) and one pillowed layer (magenta).

(Non-Uniform Rational B-Spline) or T-splines as a basis for analysis, requires models with a volumetric representation. For the sake of integration of engineering design and analysis, a fundamental step is to automatically construct solid trivariate spline models from boundary surfaces.

A few works have been devoted to construct solid spline models from boundary representations. A method was presented in [1] to generate NURBS parameterizations of swept volumes via sweeping a closed curve and isogeometric analysis was applied to the generated NURBS model. In [23], an approach was proposed to construct solid NURBS for patient-specific vascular geometric models from image data for use in isogeometric analysis. In [13], a volumetric parameterization based on discrete volumetric harmonic functions was used to construct a single trivariate B-spline. A global one-piece trivariate spline scheme was presented in [11], based on the generalized polycube parameterization [19]. There are a few volumetric parameterization methods based on harmonic functions for solid modeling applications. In [12], an automatic algorithm for computing harmonic volumetric mapping between two models of the same topology was proposed, based on the boundary mapping between the two models. A harmonic mapping method from a 3 manifold to a 3D solid sphere was developed for applications in computer graphics and medical imaging [9]. By using an adaptive tetrahedral meshing and mesh untangling technique, an algorithm was developed to construct a trivariate T-spline representation of genus-zero solids [5]. However, the boundary surface continuity of the obtained T-spline is C^0-continuous around the eight corner nodes and across the twelve edges of the parametric cube. In addition, the obtained solid T-splines have elements with negative Jacobians at the Gauss quadrature points.

This paper describes a novel method to construct solid rational T-splines for complex genus-zero geometry from the boundary surface triangulation (see the Stanford bunny model in Figure 1(a)), with C^2-continuity everywhere over the boundary surface except for the local region of only eight corner nodes and no any negative Jacobians. The definition of the rational T-spline was introduced in [21], which is used to obtain partition of unity basis functions for an arbitrary gap-free T-mesh. We first build a parametric mapping between the triangulation and the boundary of the parametric domain, a unit cube. Then an octree subdivision is carried out for the cube. In the meantime the boundary nodes are mapped to the input triangle mesh and the interior nodes are relocated via mesh smoothing. The subdivision terminates when the surface approximation error between the T-mesh and the input triangle mesh is less than a threshold. After that, we pillow one layer on the boundary, improve T-mesh quality, and implement the templates to handle extraordinary nodes and partial extraordinary nodes in order to make the T-mesh gap-free. The obtained solid T-spline is C^2-continuous except for the local region around each extraordinary node and partial extraordinary node. The boundary surface of the obtained T-spline is C^2-continuous everywhere except for the local region around only eight corner nodes. Finally, Bézier elements are extracted to facilitate T-spline based isogeometric analysis. We have applied the algorithm to several examples.

The remainder of this paper is organized as follows. Section 2 presents an overview of the construction algorithm. Then Section 3 explains T-mesh construction. Section 4 describes solid T-spline construction. Section 5 presents some T-spline results, and Section 6 draws conclusions.

2 Algorithm Overview

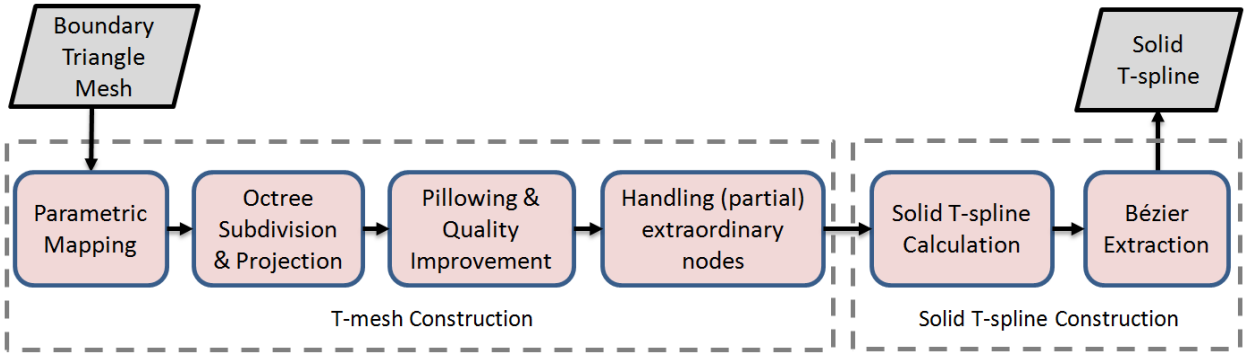


Fig. 2. An overview of the solid T-spline construction algorithm from the given boundary triangle mesh with genus-zero topology.

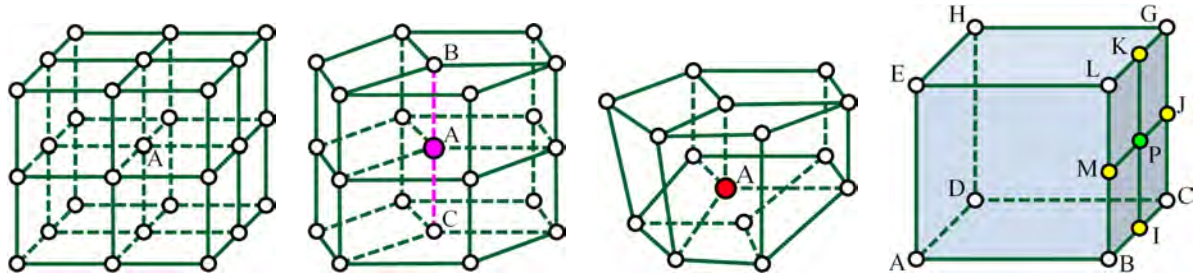
As shown in Figure 2, there are two main stages for constructing a solid T-spline from a given boundary triangle mesh. We build the T-mesh in the first stage and construct solid T-splines from the obtained T-mesh in the second stage. The T-mesh contains four different kinds of nodes: regular nodes, T-junctions, partial extraordinary nodes and extraordinary nodes. A ***regular node*** is a node about which each adjacent edge has a reflection edge, like node A in Figure 3(a). A pair of ***reflection edges*** are two adjacent edges with one common node and all the elements sharing one edge are topologically symmetric with all the elements sharing the other one. For example, AB and AC in Figure 3(b) are a pair of reflection edges. A ***partial extraordinary node*** is an irregular node about which some but not all of its adjacent edges have reflection edges. For example, node A in Figure 3(b) is a partial extraordinary node, because its two adjacent edges, AB and AC, are a pair of reflection edges while the other adjacent edges do not have reflection edges. An ***extraordinary node*** is an irregular node about which none of its adjacent edges has a reflection edge, such as node A in Figure 3(c). There are two kinds of T-junctions in 3D: edge T-junction and face T-junction. Here are their definitions:

Definition 3.1. *For solid T-splines, an **edge T-junction** is one T-junction which lies on one edge, such as nodes K, M, I and J in Figure 3(d).*

Definition 3.2. *A **face T-junction** is one T-junction which lies on one face, such as node P in Figure 3(d).*

T-mesh construction consists of the following four steps:

- Parametric Mapping - We build a parametric mapping between the input triangle mesh and a unit cube, which serves as the parameter domain for the solid T-spline;
- Octree Subdivision and Projection - The strongly balanced octree subdivision is applied for the parameter domain and each obtained node on the cube boundary is projected onto the boundary surface;
- Pillowing and Quality Improvement - We pillow all the boundary nodes and improve T-mesh quality via smoothing and optimization;



(a) (b) (c) (d)

Fig. 3. Regular node, partial extraordinary node, extraordinary node, edge T-junction and face T-junction in 3D. Node A rendered in white is a regular node (a); the magenta node is a partial extraordinary node (b); the red node are extraordinary nodes (c); yellow nodes are edge T-junctions (d); and the green node is a face T-junction (d).

- Handling Extraordinary Nodes and Partial Extraordinary Nodes - In order to make the T-mesh gap-free, we apply templates to each extraordinary node and partial extraordinary node.

Based on the valid T-mesh built from the input triangle mesh, the solid rational T-spline is constructed in the second stage. For each node in the T-mesh, the knot vectors along the three parametric directions are determined by traversing T-mesh faces and edges [16]. For each local domain, the nodes with non-zero basis functions are found and the solid T-spline is calculated based on the rational T-spline definition [21]. Bézier elements are extracted from the solid T-spline to facilitate the isogeometric analysis.

3 T-mesh construction

This stage aims to build one valid T-mesh from the given boundary triangulation. There are four main steps in the T-mesh construction stage and we will discuss each of them in detail.

3.1 Parametric Mapping

The main goal of this step is to create a parametric mapping between the boundary triangle mesh $\mathcal{T}$ and the boundary of a unit cube C, which is the parameter domain of the solid T-spline. To do this, we first select eight vertices [1] in $\mathcal{T}$, V_i $(i = 0,\ldots,7)$, which correspond to the eight corners of the cube C, C_i $(i = 0,\ldots,7)$. Then twelve curves are found via calculating the shortest distance between each pair of the selected vertices, $\overline{V_iV_j}$. The shortest distance is obtained using Dijkstra's algorithm [4], which solves the single-source shortest path problem for a graph. Based on the calculated twelve curves, we can divide the input mesh into six sub-meshes, $\mathcal{T}^i$ $(i = 0,\ldots,5)$, and

[1] In this paper, the term "vertex" connotes points in the triangle mesh and "node" means a control point in the T-mesh.

each one is associated with one face of the cube, ${\mathcal{F}_C}^i (i = 0, \ldots, 5)$. Then the main work is to map each sub-mesh $\mathcal{T}^i$ to a planar unit square ${\mathcal{F}_C}^i$ using a surface parameterization.

Surface parameterization aims at creating a one-to-one mapping f from a given surface $S \subset \mathbb{R}^3$ to a parameter domain S^*, $f(p) = q$ ($p \in S$ and $q \in S^*$). Surface parameterization has various applications in texture mapping, morphing, remeshing and data fitting [6, 7, 18]. A considerable amount of work has been done on surface parameterization and different techniques have been proposed to minimize the distortion in either angles or areas during the mapping. For a triangle mesh parameterization, the main goal is to obtain a map between the mesh and a triangulation of a domain, typically a planar domain. In other words, given a disk-like triangular mesh $\mathcal{T} \subset \mathbb{R}^3$, surface parameterization aims to find the correspondence between $\mathcal{T}$ and a simply-connected planar region, such as the unit disk or a rectangle. For planar domain parameterization [6], the surface boundary is first mapped to the boundary of the parameter domain and then the parameterization for the interior vertices is obtained by solving a linear system,

$$\sum_{j \in n_i} w_{ij}(f(V_j) - f(V_i)) = 0, \tag{1}$$

where $V_i \in S$, w_{ij} is the weight and n_i is the number of vertices adjacent to V_i. Different parameterization methods assign different weights w_{ij} for each edge. In this paper, we choose the harmonic weights,

$$w_{ij} = \cot\alpha_{ij} + \cot\beta_{ij}, \tag{2}$$

where α_{ij} and β_{ij} are the opposite angles in the two triangles shared by the edge $V_i - V_j$, as shown in Figure 4. The weights are derived using a discrete harmonic map in order to reduce the angular distortion.

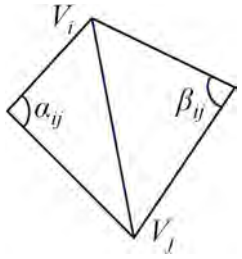
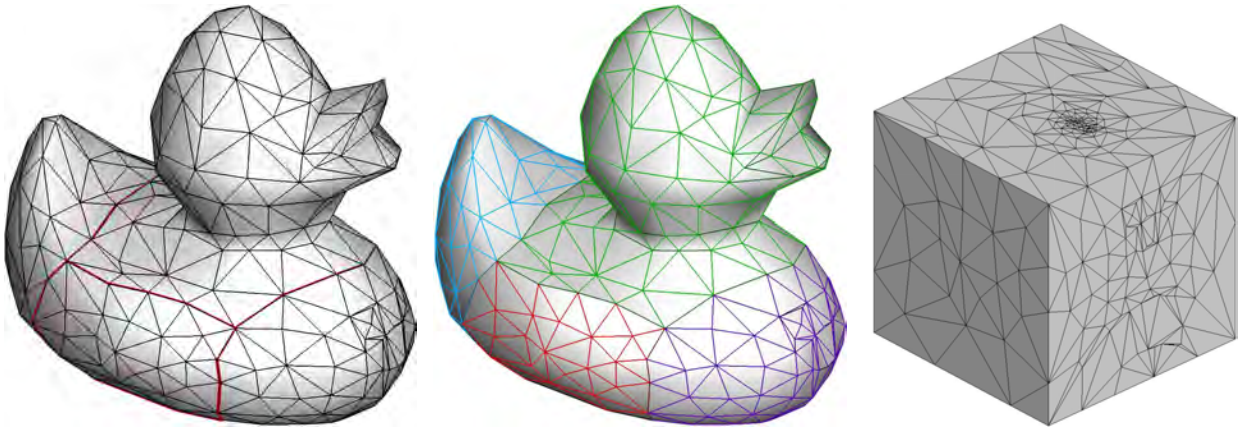


Fig. 4. Angles used for harmonic weights.

Here, each curve on the input mesh is mapped onto its corresponding edge of the cube via a chord length parameterization, in which the parameter value increases proportionally to the chord length from the start of the curve. In this way, for each sub-mesh $\mathcal{T}^i$ we have the parameterization for its boundary nodes. The parameterization for the interior vertices is calculated by solving the linear system in Equation (1). Since the associated matrix is symmetric and positive definite, the linear system is uniquely solvable and can be solved efficiently. Although the weights are not always positive in general and the obtained parameterization may not be bijective, in practice the method often gives good visual results and is one of the most popular surface parameterization methods.

Figure 5 shows one example for the mapping process. Figure 5(a) shows the shortest path calculation result, based on which we divide the input mesh into six sub-meshes as shown in Figure



(a) (b) (c)

Fig. 5. The mapping result for the duck model. (a) The input triangle mesh and the calculated shortest paths rendered in red; (b) the obtained six sub-meshes; and (c) the result after mapping the triangle mesh onto the unit cube.

5(b). Different patches are rendered in different colors. Figure 5(c) is the mapping result for the duck model and the mapping is bijective. When the mapping is not bijective, there may be some overlapping triangles. In that case, the obtained T-mesh may also overlap in some local region. This problem will be solved automatically during the following smoothing and optimization step.

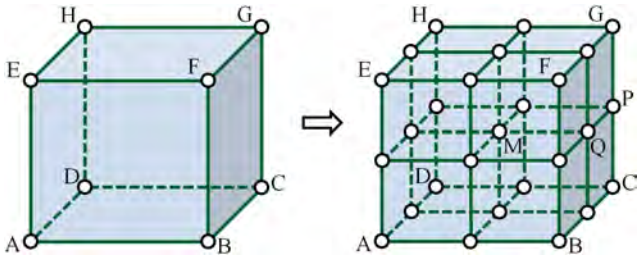


Fig. 6. Subdivision of one element into eight smaller ones.

3.2 Adaptive Octree Subdivision and Projection

In this step, we generate an adaptive initial T-mesh by applying an adaptive octree subdivision to the unit cube C, based on the mapping result in Section 3.1. Starting from the unit cube C, we subdivide one element into eight smaller ones recursively to get the refined initial T-mesh. Figure 6 shows the subdivision template. For each element lying on the boundary, we check the local distance from the T-mesh boundary to the input triangular mesh, and subdivide the element if the distance is greater than a given threshold ε. Each node C_i in the initial T-mesh has its parameter coordinates and physical coordinates. In this step, we temporarily adopt the global parameterization, hence for each node we use its coordinates in C as its parameter coordinates. For each boundary node, the physical coordinates are its mapped location on the triangular mesh. Given the parameter coordinates of one node C_i, we first loop over the triangles in the input mesh $\mathcal{T}$ to find which triangle contains the node, and then the physical coordinates are calculated using the barycentric

coordinates of C_i in this triangle. For interior edge nodes, such as node P in Figure 6, the physical coordinates are the average of the two endpoints of that edge. For interior face nodes like node Q in Figure 6, the physical coordinates are the mass center of that face. For the interior body nodes such as node M in Figure 6, the physical coordinates are the weighted average of the mass centers of all its neighboring elements. The octree subdivision continues until the local distance from each boundary element to the input triangle mesh is less than ε. Here, we adopt the strongly balanced octree subdivision, which means the level difference between two neighbouring octree elements is at most one.

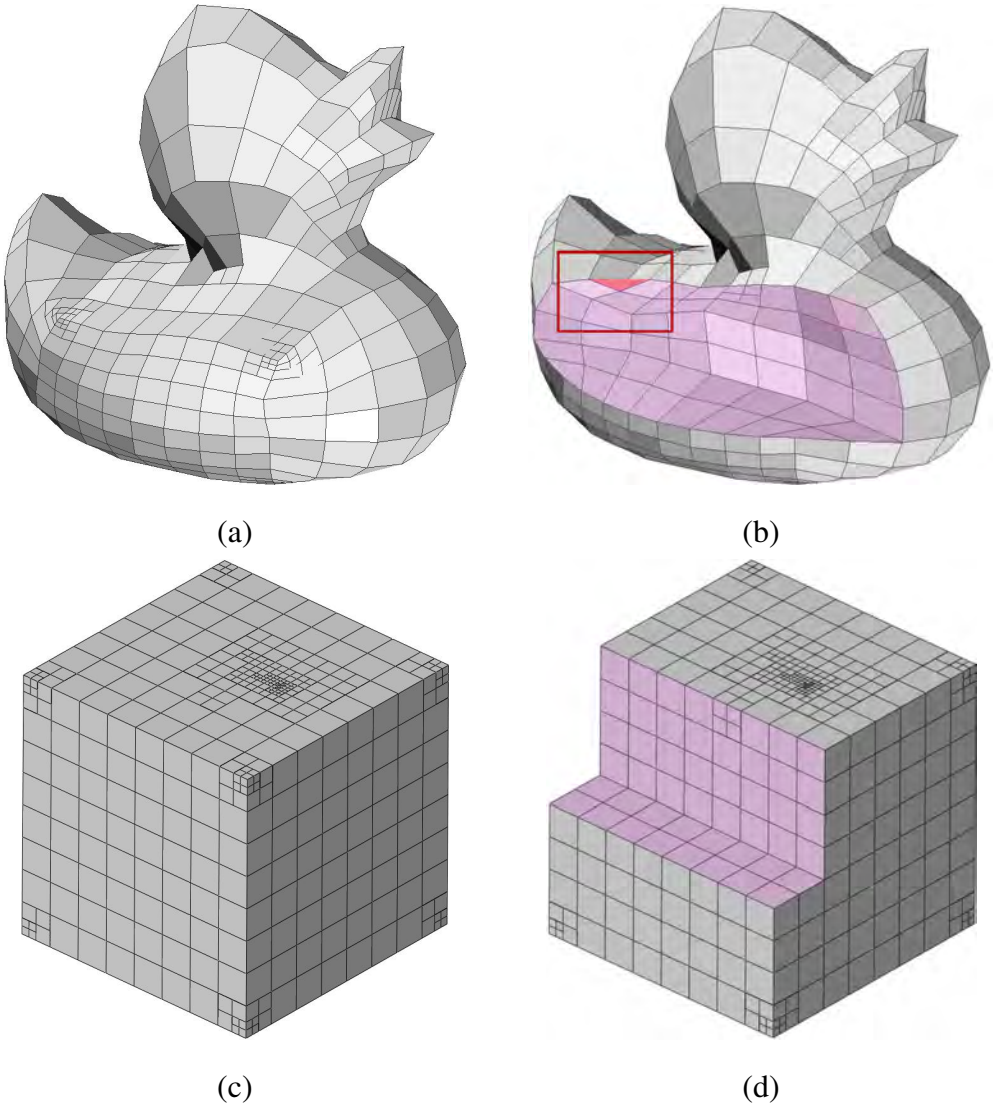


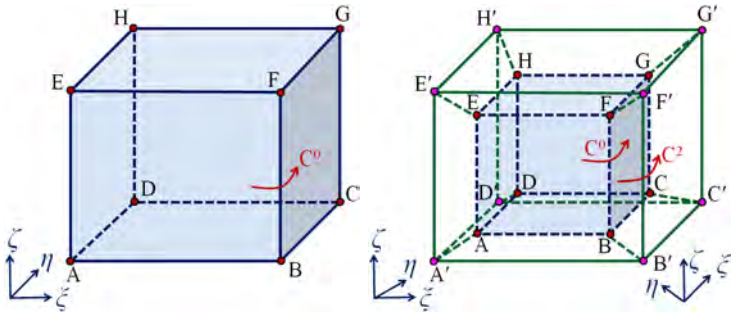
Fig. 7. The subdivision and projection results for the duck model. (a) The initial T-mesh; (c) the parameter domain; (b) and (d) are the initial T-mesh and the parametric domain with some elements removed to show the interior mesh.

Via octree subdivision, we can obtain a multi-resolution solid T-spline by choosing different octree levels. Figure 7 gives one octree subdivision and projection result for the duck model. (a) shows the obtained initial T-mesh and (c) shows the subdivision result in the parameter domain. In other words, (a) shows the physical coordinates for the nodes in the T-mesh and (b) shows their

parameter coordinates. (b) and (d) show the interior mesh with some elements removed. We can observe that there are some concave elements in the initial T-mesh which will introduce negative Jacobians, such as the red element in (b). Elements with more than two faces lying on the boundary generally have bad quality, and it is very difficult to improve their quality by relocating the nodes without losing local geometric features. To circumvent this difficulty, we adopt the pillowing technique in the following step. For complicated models, the initial T-mesh may contain some self-intersections and tangling. Hence, we also use the smoothing and optimization techniques to improve the T-mesh quality.

3.3 Pillowing and Quality Improvement

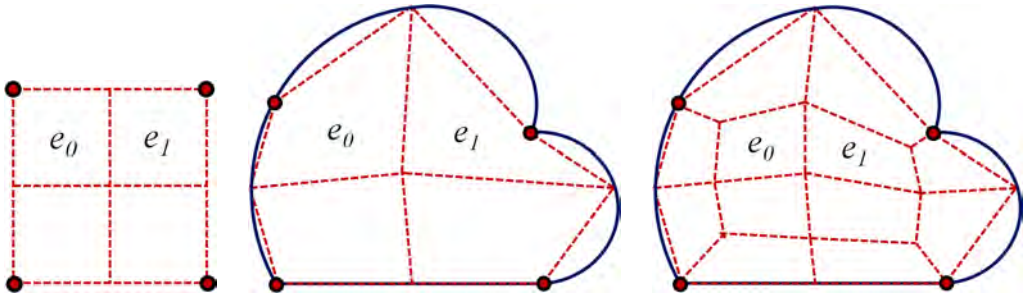
Pillowing is a sheet-insertion technique that inserts one layer around the boundary [22, 14]. Here we apply the pillowing operation to our initial T-mesh for two main purposes: (1) it allows more flexibility to improve the T-mesh quality; and (2) it helps to improve the surface continuity of solid T-splines.



(a) (b)

Fig. 8. A comparison of the T-mesh results without pillowing and with pillowing in 3D. (a) The parametric domain without pillowing; and (b) the parametric domain with pillowing.

By using the pillowing technique, we can ensure that each element has at most one face on the boundary, which gives more flexibility to further improve the mesh quality. Each boundary face is duplicated to form one pillowed element. For the pillowed element, the knot interval for the edges along the pillowing direction is a constant, and for the other two directions the knot intervals stay the same. In the pillowing step, the eight corners in the initial T-mesh (nodes $A-H$ in Figure 8(a)) become interior extraordinary nodes and the nodes on the twelve edges become interior partial extraordinary nodes. On the new boundary, there are only eight partial extraordinary nodes ($A'-H'$ in Figure 8(b)), which are pillowed from the original eight corners. Since in this step we introduce some extraordinary nodes and partial extraordinary nodes, we will use a local parameterization, instead of the global parameterization, in the following steps. Figure 9 shows one comparison of two T-mesh results without pillowing and with pillowing in 2D. Elements e_0 and e_1 in (b) have two edges on the boundary, which makes concave elements or elements of bad quality (see element e_1). After we pillow one layer, there will be no such elements and the quality can be improved dramatically as shown in (c). In addition, the pillowing technique helps to improve the surface continuity as well.



(a) (b) (c)

Fig. 9. A comparison of the T-mesh results without pillowing and with pillowing in 2D. (a) The parametric domain; (b) the T-mesh result without pillowing; and (c) the T-mesh result with pillowing.

Lemma 1 *As shown in Figure 8(a), without pillowing the surface continuity of the constructed solid T-spline is C^0 around the eight corners $A-H$ and across the twelve edges in the parameter domain, and C^2 everywhere else.*

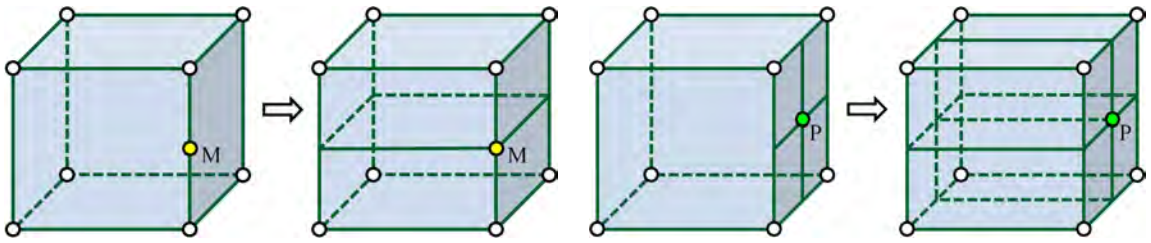
Proof: Let us take the edge BF in Figure 8(a) as an example and suppose the parametric domain is $[0,1]$ for each direction using A as the parametric origin. The surface defined by $ABFE$ is $S(\xi,0,\zeta)$ and the surface defined by $BCGF$ is $S(1,\eta,\zeta)$. The continuity across the shared boundary of two neighbouring surface patches depends on whether or not they share the same normal at the shared boundary. The surface normal of $S(\xi,0,\zeta)$ and $S(1,\eta,\zeta)$ at the common curve (edge BF) are $n^0 = \frac{\partial S(\xi,\eta,\zeta)}{\partial \xi}(1,0,\zeta) \times \frac{\partial S(\xi,\eta,\zeta)}{\partial \zeta}(1,0,\zeta)$ and $n^1 = \frac{\partial S(\xi,\eta,\zeta)}{\partial \eta}(1,0,\zeta) \times \frac{\partial S(\xi,\eta,\zeta)}{\partial \zeta}(1,0,\zeta)$, respectively. At any point $(1,0,\zeta)$ on edge BF, the two surface patches share the same control points and basis functions, but $\frac{\partial S(\xi,\eta,\zeta)}{\partial \xi}(1,0,\zeta) \neq \frac{\partial S(\xi,\eta,\zeta)}{\partial \eta}(1,0,\zeta)$. Hence, we can conclude that $n^0 \neq n^1$. In other words, the two surface patches are C^0-continuous across the shared edge. For each corner node like F, since the continuity is C^0 across each of its adjacent edges (BF, GF and EF), the surface is C^0-continuous around it. □

Let us check the surface continuity after pillowing. We define the ***p-ring neighborhood*** around one extraordinary node in 2D as follows [20]: The one-ring neighborhood is formed by all its adjacent T-mesh faces. The two-ring neighborhood is formed by the one-ring neighborhood plus all the T-mesh faces adjacent to faces on the one-ring neighborhood. This process is repeated p times to get the p-ring neighborhood.

Lemma 2 *After pillowing, the constructed T-spline surface is C^0-continuous around the eight corners $A'-H'$ until the 2-ring neighborhood, and C^2-continuous everywhere else. For the interior region, the continuity is C^0 around $A-H$ and across the twelve edges of the cube $ABCD-EFGH$.*

Proof: After pillowing, across the edges of the cube $A'B'C'D'-E'F'G'H'$, there are two domains and they have different control points and basis functions. Suppose η is along the pillowing direction and all the interior nodes have zero basis function value and zero derivatives with respect to ξ and ζ. This means that the boundary surface can be treated as one single T-spline surface, which has eight extraordinary nodes $A'-H'$. All the other nodes on the boundary are regular nodes or T-junctions. By using a local parameterization, only the extraordinary nodes or partial extraordi-

nary nodes can make the local continuity worse than C^2. Therefore, after pillowing the surface is C^0-continuous around the eight nodes $A' - H'$ until the 2-ring neighborhood and C^2-continuous everywhere else as proved in [20]. For the interior region, there are eight extraordinary nodes $A - H$ and the nodes lying on the twelve edges of cube $A - H$ are all partial extraordinary nodes. Hence, the solid T-spline is C^0-continuous around $A - H$ and also across the twelve edges, but C^2-continuous everywhere else in the interior. □



(a) (b)

Fig. 10. Refinement for the edge T-junction (a) and face T-junction (b) during T-mesh quality improvement.

Quality improvement is designed to remove tangling elements and improve the T-mesh quality by relocating T-mesh nodes via smoothing and optimization. In smoothing, each node is moved towards the mass center using its neighboring elements. In optimization, for each node we find an optimal position which maximizes the worst Jacobian. The Jacobian here is defined as

$$J = det(J_M) = \begin{vmatrix} \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} x_i \frac{\partial N_i}{\partial \zeta} \\ \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} y_i \frac{\partial N_i}{\partial \zeta} \\ \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \xi} & \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \eta} & \sum_{i=0}^{7} z_i \frac{\partial N_i}{\partial \zeta} \end{vmatrix}, \tag{3}$$

where N_i is the trilinear shape functions used in finite element analysis. The scaled Jacobian for a hexahedral T-mesh element is

$$J_s = \frac{J}{\| J_M(\cdot,0) \| \| J_M(\cdot,1) \| \| J_M(\cdot,2) \|}, \tag{4}$$

where $J_M(\cdot,0)$, $J_M(\cdot,1)$ and $J_M(\cdot,2)$ means the first, second and last column of the Jacobian Matrix J_M, respectively. However, the smoothing and optimization techniques work only for unstructured hexahedral meshes without hanging nodes or T-junctions. To handle T-junctions in our T-mesh, we refine the local region to convert the local T-mesh to a hexahedral mesh virtually. For the edge T-junction node, we treat the element as two hexahedral elements as shown in Figure 10(a). For the face T-junction node, we treat the element as four hexahedral elements as shown in Figure 10(b). Then for each node, we loop over all the adjacent virtual "hexahedral" elements, identify the element with the worst Jacobian and then relocate the node to maximize the worst Jacobian. Figure 11 shows the result after pillowing and optimization for the duck model. It is obvious that the T-mesh quality becomes much better (see Table 1 in Section 5).

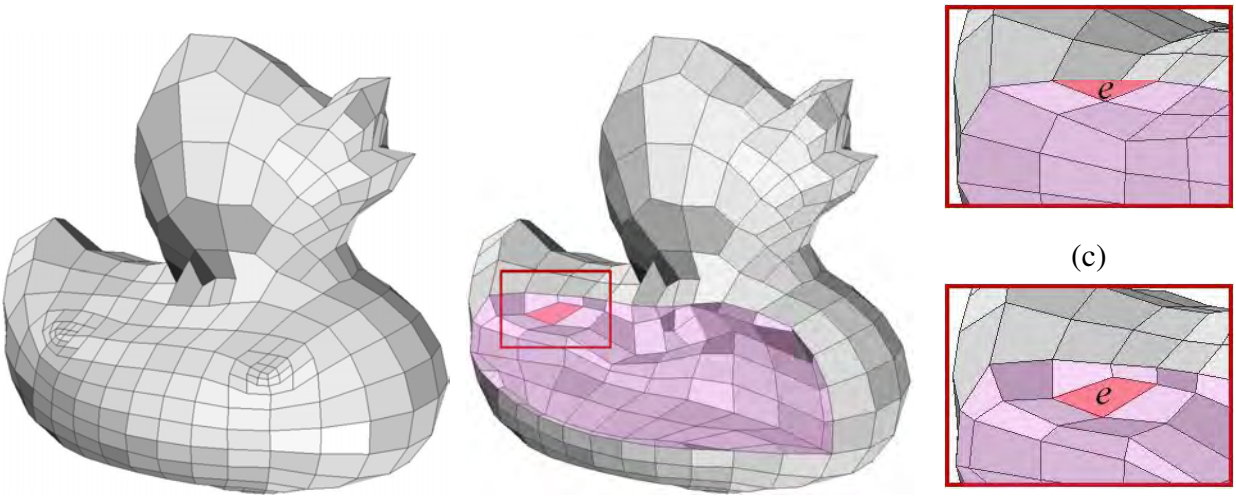


Fig. 11. The pillowing and optimization results for the duck model. (a) The T-mesh after pillowing and optimization; (b) The interior of the T-mesh; (c) and (d) are zoom-in pictures of one local region before and after pillowing and optimization. (c) is the same local region in Figure 7(b).

3.4 Handling extraordinary nodes and partial extraordinary nodes

The extraordinary nodes or partial extraordinary nodes may introduce gaps to the solid T-spline. This step aims to make the initial T-mesh gap-free by designing templates for each type of node and applying them to elements. Figure 12(a) shows the general template for a partial extraordinary node. The magenta edge adjacent to the partial extraordinary node has a reflection edge about this node. Three templates were given for a partial extraordinary node in [21]. Here, since the T-mesh obtained in this paper is not quasi-uniform, we choose the template in Figure 12(a) which works for a general T-mesh. Figure 12(b) shows one general template for an extraordinary node. These templates can guarantee the obtained T-mesh is gap-free as proved in [20, 21]. For the obtained T-mesh in this paper, in the interior region there are eight extraordinary nodes $A-H$ and the nodes lying on the twelve edges of the cube $ABCD-EFGH$ are partial extraordinary nodes. The solid T-spline is C^0-continuous around $A-H$ and across the twelve edges. On the boundary, there are eight partial extraordinary nodes, which make their surrounding regions C^0-continuous. Everywhere else is C^2-continuous.

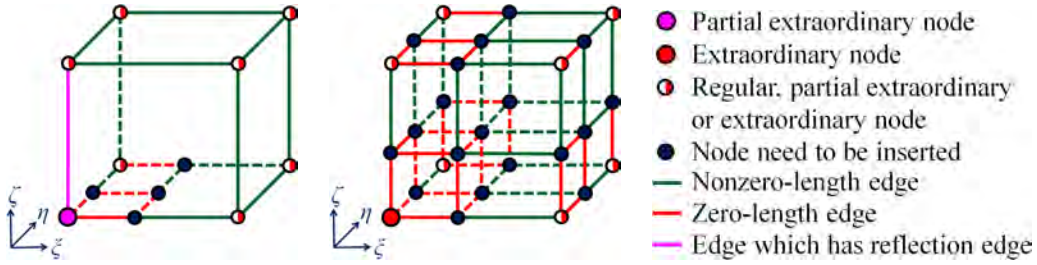


Fig. 12. The general template for partial extraordinary nodes (a) and extraordinary nodes (b).

4 Solid T-spline Construction and Bézier Extraction

4.1 Solid T-spline Construction

In this step, we aim to build the rational solid T-spline from the T-mesh obtained in the first stage. T-splines were introduced in [17], which allow T-junctions, L-junctions in their control grid and local refinement [16]. Rational T-splines were generalized from T-splines, in order to obtain basis functions satisfying a partition of unity [21]. The rational solid T-spline is defined as

$$S(\xi,\eta,\zeta) = \frac{\sum_{i=0}^{n} w_i C_i R_i(\xi,\eta,\zeta)}{\sum_{i=0}^{n} w_i R_i(\xi,\eta,\zeta)}, \quad (\xi,\eta,\zeta) \in \Omega, \tag{5}$$

where

$$R_i(\xi,\eta,\zeta) = \frac{N_i^{\xi}(\xi) N_i^{\eta}(\eta) N_i^{\zeta}(\zeta)}{\sum_{j=0}^{n} N_j^{\xi}(\xi) N_j^{\eta}(\eta) N_j^{\zeta}(\zeta)} \tag{6}$$

is the rational B-spline basis function, N_i^{ξ}, N_i^{η} and N_i^{ζ} are B-spline basis functions defined by the local knot vectors at node C_i which, for degree $d = 3$, are given by $\vec{\xi}_i = [\xi_{i0}, \xi_{i1}, \xi_{i2}, \xi_{i3}, \xi_{i4}]$, $\vec{\eta}_i = [\eta_{i0}, \eta_{i1}, \eta_{i2}, \eta_{i3}, \eta_{i4}]$ and $\vec{\zeta}_i = [\zeta_{i0}, \zeta_{i1}, \zeta_{i2}, \zeta_{i3}, \zeta_{i4}]$. Obviously the summation of all the rational B-spline basis functions is always 1, or we have $\sum_{j=0}^{n} R_i = 1$ for any (ξ,η,ζ). The knot vectors for each node can be inferred from the T-mesh by traversing T-mesh faces and edges, and the rational basis functions are built based on the knot vectors.

Since we have extraordinary nodes and partial extraordinary nodes in the T-mesh, we have to use the local parameterization. In other words, each domain has its own parametric coordinate system and a node may have different knot vectors in different local domains. Therefore, a naive way to build the solid T-spline from the obtained T-mesh is to

- for each domain find all the nodes on the 2-ring neighborhood, which may have non-zero basis functions in it;
- obtain the knot vectors of these nodes by traversing T-mesh faces and edges using the local parametric coordinate system; and then
- build the rational basis functions and the local solid T-spline element.

In this way, for each node we repeat traversing T-mesh faces and edges by shooting rays from the node many times, which is time-consuming. To overcome this problem, we observe that the knot vectors of a regular node, T-junction or L-junction always share the same knot intervals, no matter which local domain we are considering. The knot vectors of a node in one domain can be obtained from the knot vectors in another domain by a coordinate system transformation. Here, we take advantage of this property to improve the algorithm. For regular nodes, T-junctions or L-junctions

we first calculate the knot intervals by traversing T-mesh faces and edges using the parametric coordinate system of its first neighboring element as the reference. For each domain, we

- loop over each node C_i on the 2-ring neighborhood and obtain the knot coordinates $(\xi_{i2}, \eta_{i2}, \zeta_{i2})$ of C_i based on the local parametric coordinate system;
- check the relationship between this local parametric coordinate system and the node's reference system, and calculate the knot vectors based on the coordinate system transformation;
- determine all the nodes which have non-zero basis functions, and use them to build rational basis functions and the local solid T-spline element.

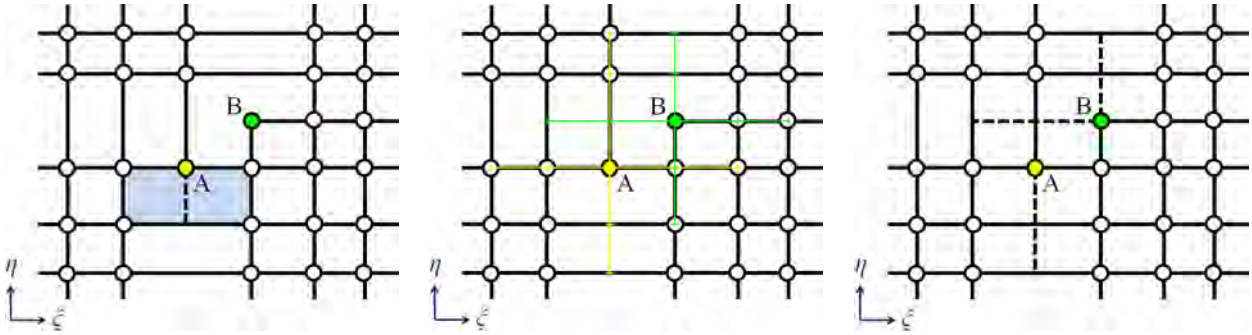
In this way, we avoid repeating calculation of knot intervals or knot vectors by traversing T-mesh faces and edges. However, for partial extraordinary nodes and extraordinary nodes, the knot intervals may not be the same for different domains. Hence we have to recalculate their knot intervals each time. The whole solid T-spline model is built by looping over all the local domains and constructing the local solid T-spline elements.

4.2 Bézier Extraction

To facilitate isogeometric analysis, we extract Bézier elements from the constructed solid T-spline [3, 15], which serve as the primary computational objects. Generally, one element in the T-mesh may contain more than one Bézier elements and the T-mesh itself does not fully delineate the reduced continuity lines or knot lines in the parametric space. For example, the blue region in the local T-mesh shown in Figure 13(a) has one reduced continuity line (the black dashed line) due to node A. In other words, the blue element contains two Bézier elements. Hence, in order to extract the Bézier elements, we first need to determine the "Bézier mesh", which can capture all the knot lines. Then the question arises: how to obtain the "Bézier mesh" from a given T-mesh? Since all the reduced continuity lines which are not delineated in the T-mesh are introduced by T-junctions or L-junctions, we can use the knot vector inference lines to capture all these reduced continuity lines. Let us take the local T-mesh shown in Figure 13(a) as an example, which contains one T-junction A and one L-junction B. The yellow lines in Figure 13(b) are the knot vector inference lines for node A following the knot vector inference rule [16] and the green ones are the knot vector inference lines for node B. The Bézier mesh is obtained by including all these knot vector inference lines for T-junctions and L-junctions, as shown in Figure 13(c).

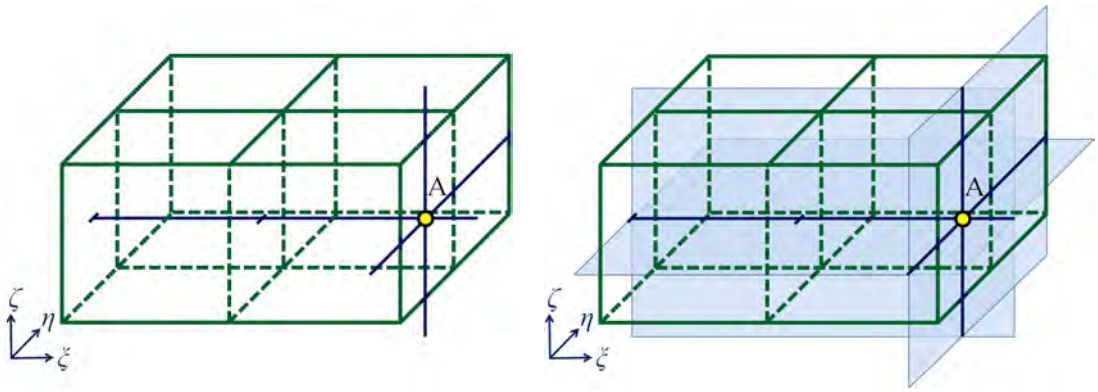
For T-meshes in 3D, the three isoparametric planes, passing through the knot vector inference lines of each T-junction or L-junction and bounded by the end knots, are used to refine the T-mesh to obtain the Bézier mesh. These planes indicate all the reduced continuity surfaces which are not delineated in the T-mesh. The Bézier mesh is obtained after adding them to the T-mesh. Let us take one local T-mesh shown in Figure 14(a) as an example, which contains one face T-junction A. The blue lines in Figure 14(a) are the knot vector inference lines for node A and the blue planes are the isoparametric planes with the knot vector inference lines at A and bounded by the end knots. The Bézier mesh is obtained after refining the T-mesh using these isoparametric planes.

Each element in the Bézier mesh corresponds to one Bézier element. The next step is to calculate



(a) (b) (c)

Fig. 13. Inferring the Bézier mesh from a T-mesh in 2D. (a) A local region of a T-mesh with one T-junction A and one L-junction B; (b) the T-mesh with all the knot vector inference lines for nodes A and B; and (c) the Bézier mesh.



(a) (b)

Fig. 14. Inferring the Bézier mesh from a T-mesh with one face T-junction in 3D. (a) The T-mesh with all the knot vector inference lines for nodes A; and (b) the Bézier mesh.

the transformation matrix M^e between the T-spline basis functions and the Bézier basis functions for each Bézier element. For a solid T-spline, we have

$$B_t^e = M^e B_b^e, \tag{7}$$

where

$$B_t^e = \left[N_0^\xi N_0^\eta N_0^\zeta, N_1^\xi N_1^\eta N_1^\zeta, \cdots, N_{n^e-1}^\xi N_{n^e-1}^\eta N_{n^e-1}^\zeta\right]^T \tag{8}$$

is the vector formed by the nonzero T-spline basis functions,

$$B_b^e = \begin{bmatrix} N[0,0,0,0,1](\xi)N[0,0,0,0,1](\eta)N[0,0,0,0,1](\zeta) \\ N[0,0,0,1,1](\xi)N[0,0,0,0,1](\eta)N[0,0,0,0,1](\zeta) \\ N[0,0,1,1,1](\xi)N[0,0,0,0,1](\eta)N[0,0,0,0,1](\zeta) \\ \vdots \\ N[0,0,1,1,1](\xi)N[0,1,1,1,1](\eta)N[0,1,1,1,1](\zeta) \\ N[0,1,1,1,1](\xi)N[0,1,1,1,1](\eta)N[0,1,1,1,1](\zeta) \end{bmatrix} \tag{9}$$

is the vector formed by the Bézier basis functions, and n^e is the number of nodes with nonzero basis function values in this domain. M^e can be calculated using the Oslo knot insertion algorithm [8]. Due to M^e, we only need to deal with the Bézier basis functions and avoid the troublesome work to infer the knot vectors for each node and calculate its basis functions in the computation. One thing we need to mention is that the basis functions used in isogeometric analysis should also be the rational basis functions. Then for one Bézier element S^e, the Jacobian matrix is defined as (for the sake of simplicity, we suppose all $w_i = 1$ in Equation (5)),

$$J^e = \left[\frac{\partial S^e}{\partial \xi}, \frac{\partial S^e}{\partial \eta}, \frac{\partial S^e}{\partial \zeta}\right] = \begin{bmatrix} \sum_{i=0}^{n^e} C_i^x \frac{\partial R_i}{\partial \xi} & \sum_{i=0}^{n^e} C_i^x \frac{\partial R_i}{\partial \eta} & \sum_{i=0}^{n^e} C_i^x \frac{\partial R_i}{\partial \zeta} \\ \sum_{i=0}^{n^e} C_i^y \frac{\partial R_i}{\partial \xi} & \sum_{i=0}^{n^e} C_i^y \frac{\partial R_i}{\partial \eta} & \sum_{i=0}^{n^e} C_i^y \frac{\partial R_i}{\partial \zeta} \\ \sum_{i=0}^{n^e} C_i^z \frac{\partial R_i}{\partial \xi} & \sum_{i=0}^{n^e} C_i^z \frac{\partial R_i}{\partial \eta} & \sum_{i=0}^{n^e} C_i^z \frac{\partial R_i}{\partial \zeta} \end{bmatrix}, \tag{10}$$

where C_i^x, C_i^y and C_i^z are the coordinates of control point C_i and

$$R_i = \frac{\sum_{j=0}^{63} M^e[i,j] B_b^e[j]}{\sum_{k=0}^{63} \sum_{j=0}^{63} M^e[k,j] B_b^e[j]}. \tag{11}$$

In this paper, we choose the scaled Jacobian J_s^e as one quality metric to measure the quality of the extracted Bézier elements,

$$J_s^e = \frac{det(J^e)}{\| \frac{\partial S^e}{\partial \xi} \| \| \frac{\partial S^e}{\partial \eta} \| \| \frac{\partial S^e}{\partial \zeta} \|}. \tag{12}$$

5 Results

We have applied the construction algorithm to several models (Figures 1, 15-18). The output solid T-spline is tricubic and C^2-continuous except in the vicinity of partial extraordinary and extraordinary nodes. From the Bézier extraction results, we can observe that the surface is very smooth except at the eight corner nodes. Statistics for all the tested models are shown in Table 1. The T-mesh Jacobian is calculated at the eight corners of one element using Equation (4). The Bézier Jacobian is calculated using the scaled Jacobian in Equation (12) at the eight Gauss quadrature points for each Bézier element. The time used includes the T-mesh construction and solid T-spline construction, but the Bézier extraction and Bézier Jacobian evaluation are not included. The algorithm is efficient and all the results were computed on a PC equipped with an Intel X3470 processor and 8GB main memory. The most time-consuming part is the mapping and T-mesh quality improvement, hence the time used for each model mostly depends on the input mesh size, the T-mesh size and the distortion introduced during the parametric mapping.

Table 1. Statistics of all the tested models

Model	Input mesh (vertices, elements)	T-mesh nodes#	T-mesh Jacobian (worst, best)	Bézier elements#	Bézier Jacobian (worst, best)	Time (s)
Duck	(319, 634)	4,794	(0.08, 1.00)	3,636	(0.01, 1.00)	8.63
Fish	(2,674, 5,344)	10,571	(0.04, 1.00)	14,837	(0.01, 1.00)	46.13
Egea	(4,500, 8,996)	9,296	(0.08, 1.00)	10,124	(0.03, 1.00)	48.56
Bunny	(4,833, 9,662)	7,664	(0.07, 1.00)	9,312	(0.02, 1.00)	79.19
Cow	(3,661, 7,318)	16,568	(0.01, 1.00)	23,943	(0.01, 1.00)	174.93

6 Conclusions

We have developed a robust and efficient algorithm to construct solid T-splines for genus-zero geometry from a boundary triangulation. The solid Bézier elements are extracted in order to facilitate isogeometric analysis. In the T-mesh quality improvement proposed in this paper, for the sake of simplicity, we treat each element as piecewise-linear to relocate the control points. Even all the T-mesh elements have positive Jacobian, there is no guarantee all the Bézier elements have positive Jacobian. In the future, we intend to develop an optimization method using the Jacobian of the Bézier elements as the objective function to relocate the control points. In addition, we intend to work on objects with arbitrary topology in our future work.

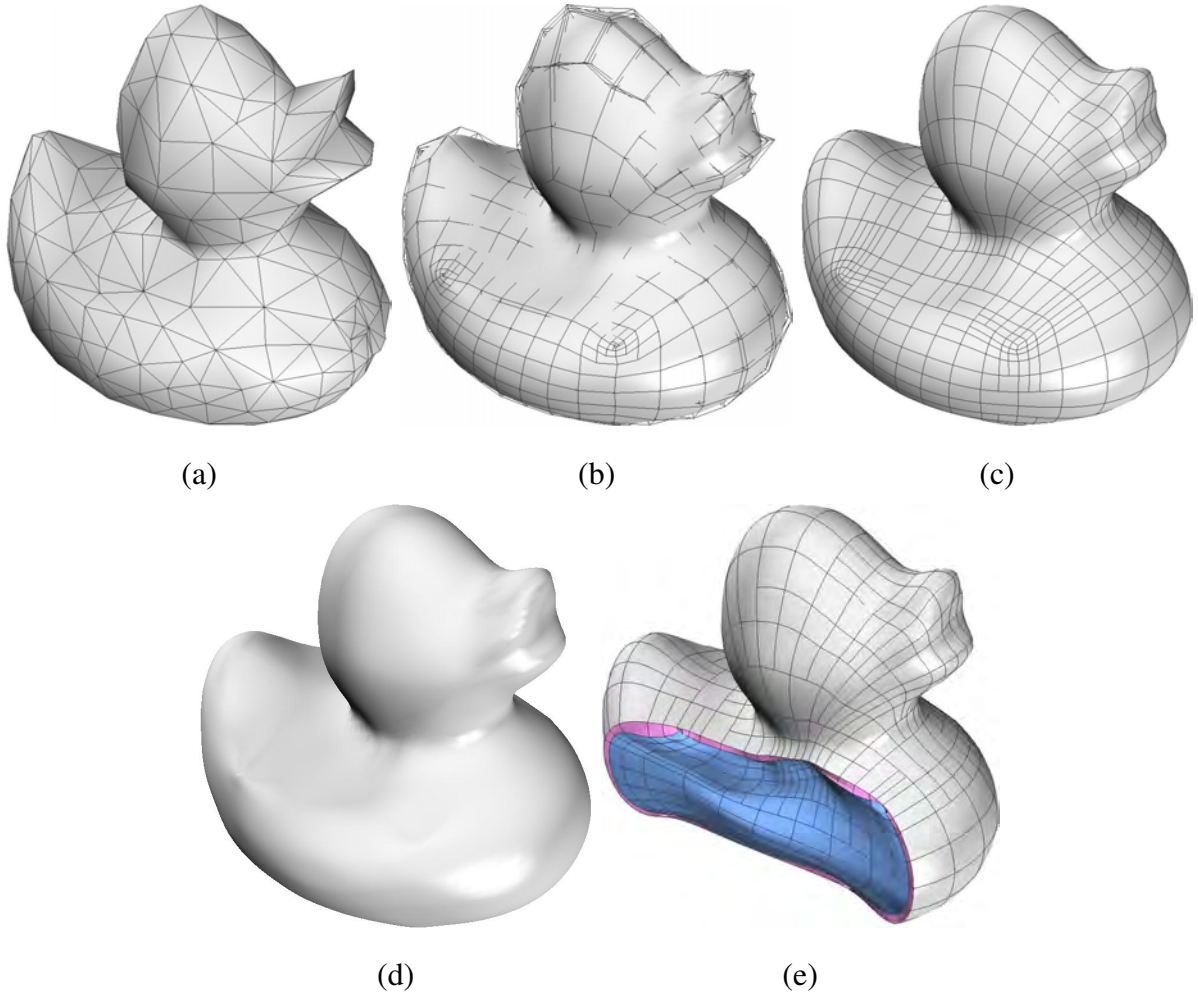


Fig. 15. The duck model. (a) The input boundary triangle mesh; (b) the constructed solid T-spline and T-mesh; (c) the extracted solid Bézier elements; (d) the solid T-spline; and (e) the extracted solid Bézier elements with some elements removed to show the interior mesh (blue) and one pillowed layer (magenta).

Acknowledgements

Y. Zhang and W. Wang were supported in part by ONR Grant N00014-08-1-0653. T. J.R. Hughes was supported by ONR Grant N00014-08-1-0992, NSF GOALI CMI-0700807/0700204, NSF CMMI-1101007 and a grant from SINTEF. The bunny model is from the Stanford Computer Graphics Laboratory. The duck and Egea models are provided by the AIM@SHAPE Shape Repository.

References

[1] M. Aigner, C. Heinrich, B. Jüttler, E. Pilgerstorfer, B. Simeon, and A. V. Vuong. Swept volume parameterization for isogeometric analysis. In *Proceedings of the 13th IMA International Conference on Mathematics of Surfaces XIII*, pages 19–44, 2009.

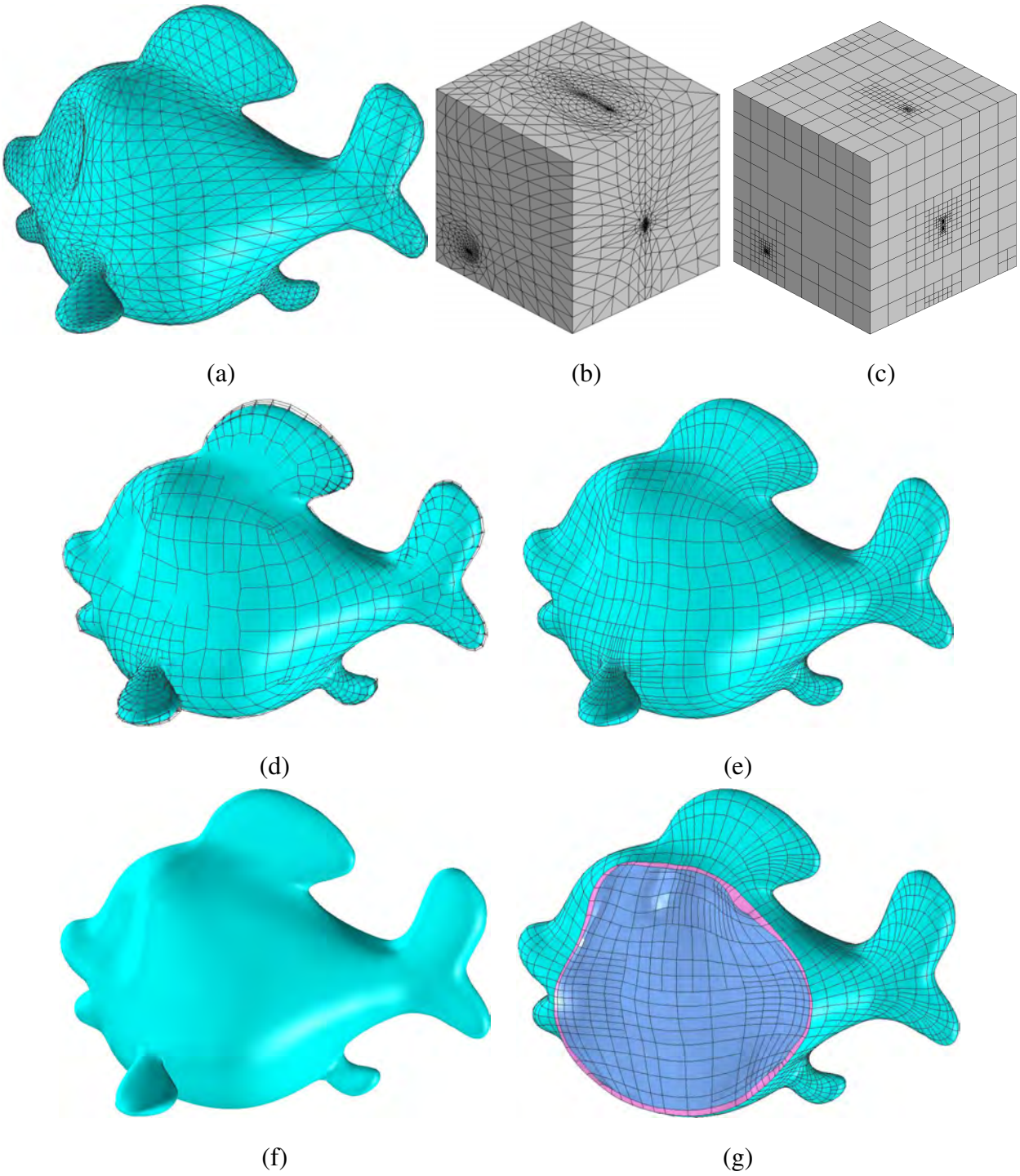


Fig. 16. Fish model. (a) The input boundary triangle mesh; (b) the mapping result; (c) the subdivision result for the parametric domain; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; (f) the solid T-spline; and (g) the extracted solid Bézier elements with some elements removed to show the interior mesh (blue) and one pillowed layer (magenta).

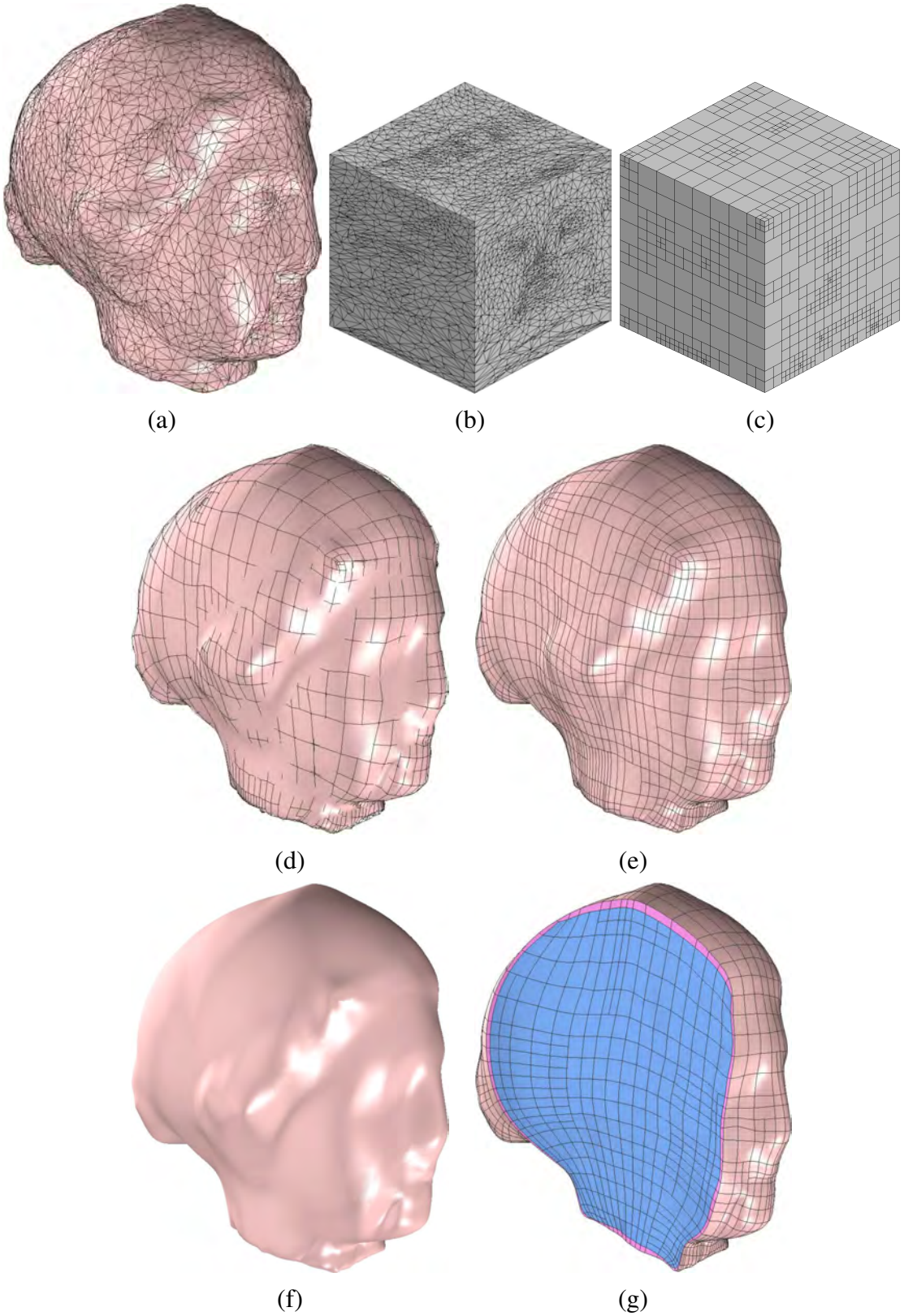


Fig. 17. Egea model. (a) The input boundary triangle mesh; (b) the mapping result; (c) the subdivision result for the parametric domain; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; (f) the solid T-spline; and (g) the extracted solid Bézier elements with some elements removed to show the interior mesh (blue) and one pillowed layer (magenta).

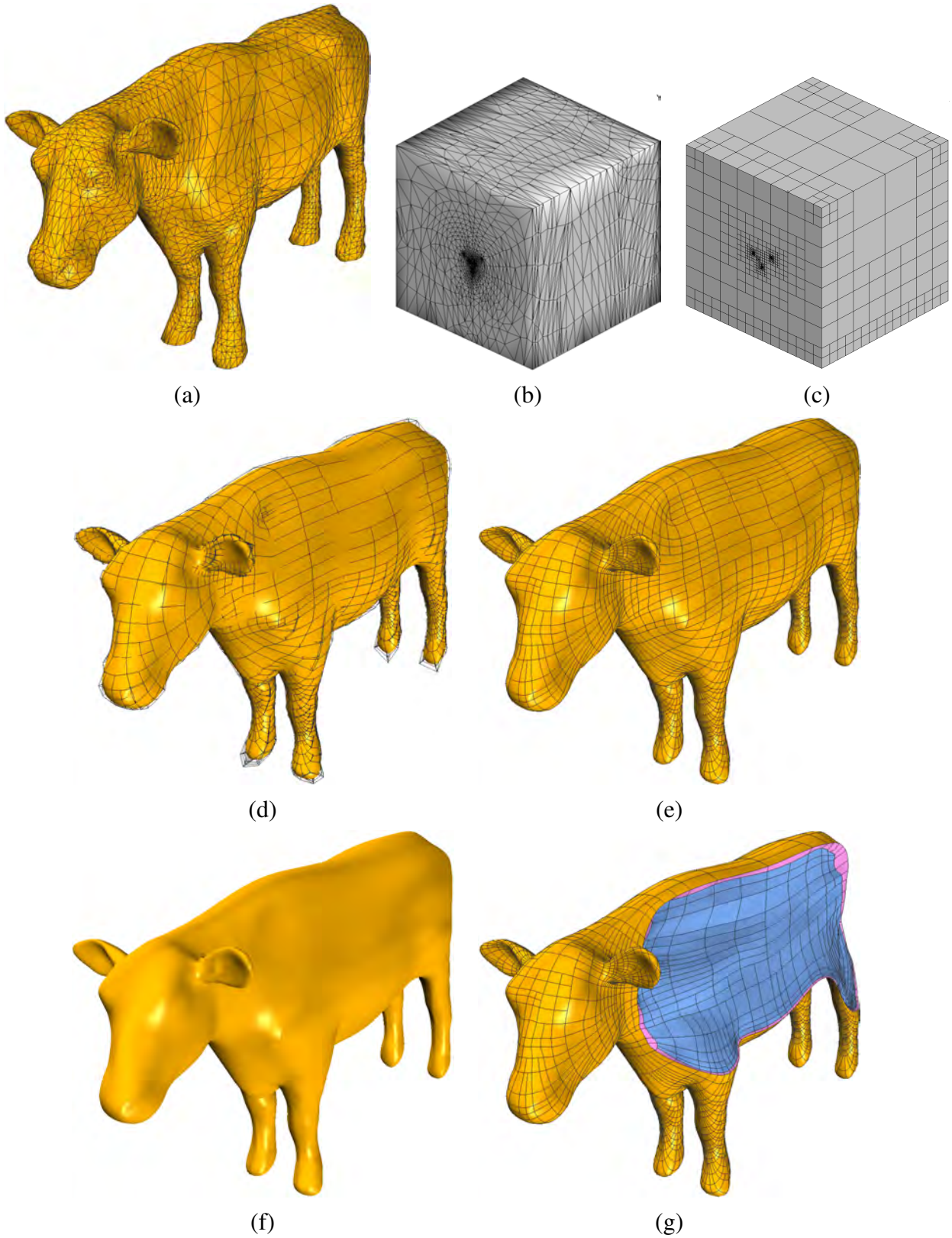


Fig. 18. The cow model. (a) The input boundary triangle mesh; (b) the mapping result; (c) the subdivision result for the parametric domain; (d) the constructed solid T-spline and T-mesh; (e) the extracted solid Bézier elements; (f) the solid T-spline; and (g) the extracted solid Bézier elements with some elements removed to show the interior mesh (blue) and one pillowed layer (magenta).

[2] Y. Bazilevs, V. M. Calo, J. A. Cottrell, J. A. Evans, T. J.R. Hughes, S. Lipton, M. A. Scott, and T. W. Sederberg. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):229–263, 2010.

[3] M. J. Borden, M. A. Scott, J. A. Evans, and T. J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87:15–47, 2011.

[4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[5] J. M. Escobar, J. M. Cascón, E. Rodríguez, and R. Montenegro. A new approach to solid modeling with trivariate T-splines based on mesh optimization. *Computer Methods in Applied Mechanics and Engineering, accepted*, 2011.

[6] M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231 – 250, 1997.

[7] M. S. Floater and K. Hormann. Surface parameterization: a tutorial and survey. *Advances in Multiresolution for Geometric Modelling*, pages 157–186, 2005.

[8] R. Goldman and T. Lyche. *Knot insertion and deletion algorithms for B-spline curves and surfaces*. Society for Industrial and Applied Mathematics–Philadelphia, 1993.

[9] X. Gu, Y. Wang, and S. Yau. Volumetric harmonic map. *Communications in Information and Systems*, 3(3):191–202, 2003.

[10] T. J.R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[11] B. Li, X. Li, K. Wang, and H. Qin. Generalized polycube trivariate splines. In *Shape Modeling International Conference*, pages 261–265, 2010.

[12] X. Li, X. Guo, H. Wang, Y. He, X. Gu, and H. Qin. Harmonic volumetric mapping for solid modeling applications. In *ACM symposium on Solid and physical modeling*, pages 109–120, 2007.

[13] T. Martin, E. Cohen, and R. M. Kirby. Volumetric parameterization and trivariate B-spline fitting using harmonic functions. *Computer Aided Geometric Design*, 26(6):648–664, 2009.

[14] J. Qian, Y. Zhang, W. Wang, A. C. Lewis, M.A. S. Qidwai, and A. B. Geltmacher. Quality improvement of non-manifold hexahedral meshes for critical feature determination of microstructure materials. *International Journal for Numerical Methods in Engineering*, 82:1406–1423, 2010.

[15] M. A. Scott, M. J. Borden, C. V. Verhoosel, T. W. Sederberg, and T. J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 87:15–47, 2011.

[16] T. W. Sederberg, D. L. Cardon, G. T. Finnigan, N. S. North, J. Zheng, and T. Lyche. T-spline simplification and local refinement. In *ACM SIGGRAPH*, pages 276–283, 2004.

[17] T. W. Sederberg, J. Zheng, A. Bakenov, and A. Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22(3):477–484, 2003.

[18] A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis.*, 2:105–171, 2006.

[19] H. Wang, Y. He, X. Li, X. Gu, and H. Qin. Polycube splines. In *Symposium on Solid and Physical Modeling*, pages 241–251, 2007.

[20] W. Wang, Y. Zhang, M. A. Scott, and T. J.R. Hughes. Converting an unstructured quadrilat-

eral mesh to a standard T-spline surface. *Computational Mechanics*, 48:477–498, 2011.

[21] W. Wang, Y. Zhang, G. Xu, and T. J.R. Hughes. Converting an unstructured quadrilateral/hexahedral mesh to a rational T-spline. *Computational Mechanics, submitted*, 2011.

[22] Y. Zhang, C. Bajaj, and G. Xu. Surface smoothing and quality improvement of quadrilateral/hexahedral meshes with geometric flow. *Communications in Numerical Methods in Engineering*, 25:1–18, 2009.

[23] Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, and T. J.R. Hughes. Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. *Computer Methods in Applied Mechanics and Engineering*, 196(29-30):2943–2959, 2007.